フルヴィッツ数を計算するアルゴリズム

片山 潤哉

2016 年 1 月 6 日

1　はじめに

フルヴィッツ数とは Hurwitz が行ったベルヌーイ数の一般化についての研究の成果である．彼は, 次の式

$$\frac{1}{\sin^2 x} = \frac{1}{x^2} + \sum_{n=2}^{\infty} \frac{(-1)^{\frac{n}{2}-1} 2^n B_n}{n} \frac{x^{n-2}}{(n-2)!}$$

において, ベルヌーイ数は三角関数の展開係数であるという見方から, これをレムニスケート関数に置き換えてもその展開係数は似たような数論的性質を持つに違いないと考え, 次の式を示した (詳細は後に述べる).

$$\wp(z) = \frac{1}{z^2} + \sum_{n=2}^{\infty} \frac{2^n H_n}{n} \frac{z^{n-2}}{(n-2)!}.$$

関数の展開係数となっている数列 H_n がフルヴィッツ数と呼ばれるものである．本研究はそのフルヴィッツ数を従来のものより高速に計算するアルゴリズムについて研究した．新しいアルゴリズム (アルゴリズム 2) において, アイゼンシュタイン級数の計算と, フルヴィッツの定理を用い分母を計算することで, 次の主結果が得られた.

アルゴリズム 2 はフルヴィッツ数 H_k をビット演算量 $O(M(k \log k)k(\log k)^2)$ で計算する.
また従来の漸化式を利用して計算するアルゴリズムはフルヴィッツ数 H_k をビット演算量 $O(M(k \log k)k^2 \log k)$ で計算する.
以上よりアルゴリズム 2 の方がより高速にフルヴィッツ数 H_k を計算する.

また, 数値計算の速度の比較実験を行った．その結果から十分大きい自然数 k に対してフルヴィッツ数 H_k を計算するのは本研究で得られたアルゴリズムの方が高速に計算できるという結果が得られた.

本論文の構成は次のようになる．第 2 節でフルヴィッツ数とベルヌーイ数の定義, 諸性質について述べる．第 3 節ではベルヌーイ数を計算するアルゴリズムを紹介し, そのアルゴリズムを基にして構成したフルヴィッツ数を計算するアルゴリズムを第 4 節で述べ, その正当性を証明する．第 5 節ではフルヴィッツ数を計算するアルゴリズムの計算量を評価し, 従来の漸化式を用いて計算するアルゴリズムよりも計算量が減っていることを示す．第 6 節にて数値実験の結果を載せた．以上が本論文の構成である.

2　フルヴィッツ数とベルヌーイ数

この節では [1] に従って, ベルヌーイ数とフルヴィッツ数について説明する.

2.1　ベルヌーイ数

この節ではベルヌーイ数について説明する. ベルヌーイ数 B_n とは次のように形式的べき級数による母関数表示

$$\frac{te^t}{e^t-1}=\sum_{n=0}^{\infty}B_n\frac{t^n}{n!}$$

によって定義される有理数のことである. 具体的な値は次の表のようになる.

n	0	1	2	3	4	5	6	7	8	9	10
B_n	1	$\frac{1}{2}$	$\frac{1}{6}$	0	$-\frac{1}{30}$	0	$\frac{1}{42}$	0	$-\frac{1}{30}$	0	$\frac{5}{66}$

またベルヌーイ数の分母は次に述べるクラウゼン・フォンシュタウトの定理によって完全に決定することができる.

定理 2.1 (クラウゼン・フォンシュタウトの定理)**.** $n=1$ および各偶数 $n\geq 2$ に対し, B_n は

$$B_n=-\sum_{p-1|n}\frac{1}{p}+C_n\ (C_n\text{は整数})$$

の形に書ける. ここでは和は $p-1$ が n を割るような素数 p をわたる.

この定理よりベルヌーイ数 B_n の分母は

$$\prod_{p-1|n}p$$

とかけることがわかる. また, $\zeta(s)=\sum_{n=1}^{\infty}\frac{1}{n^s}$ とすると各偶数 $2k$(k は自然数）に対して次の等式が成り立っていることが知られている.

定理 2.2.

$$\zeta(2k)=\frac{(-1)^{k-1}}{2}\frac{B_{2k}}{(2k)!}(2\pi)^{2k}.$$

これらの定理を用いることでベルヌーイ数を計算するアルゴリズムが構成される.

2.2　フルヴィッツ数

次にこの節ではフルヴィッツ数の定義からはじめる. そのために, まず

$$\varpi=2\int_0^1\frac{1}{\sqrt{1-x^4}}dx$$

とおく. これはレムニスケートの周長の半分であって, レムニスケート周率とも呼ばれ, その値は 2.62205…… と無限に続く. 次に, ϖ と $\varpi\sqrt{-1}$ を周期に持つワイエルシュトラスの楕円関数を

$\wp(z)$ と書く. これは無限級数で次のように定義される.

$$\wp(z) = \frac{1}{z^2} + \sum_{\substack{\lambda \in \mathbb{Z}\varpi + \mathbb{Z}\varpi\sqrt{-1} \\ \lambda \neq 0}} \left(\frac{1}{(z-\lambda)^2} - \frac{1}{\lambda^2} \right).$$

$\wp(z)$ は, 複素平面において, 格子 $\mathbb{Z}\varpi + \mathbb{Z}\varpi\sqrt{-1}$ 上の点でのみ 2 位の極をもちそれ以外では正則な複素関数で, z を $z+\lambda(\lambda \in \mathbb{Z}\varpi + \mathbb{Z}\varpi\sqrt{-1})$ に置き換えても値が変わらないという 2 重周期性をもつ. この格子が $\sqrt{-1}$ 倍で保たれるということから, $\lambda \in \mathbb{Z}[\sqrt{-1}]$ に対して $\wp(\lambda z)$ が $\wp(z)$ と $\wp'(z)$ の有理式で表されることがわかる. この性質から, $\wp(z)$ は $\mathbb{Z}[\sqrt{-1}]$ を虚数乗法にもつ楕円関数であるといわれる.

定義からわかるように $\wp(z)$ は偶関数であり, また, z, λ をともに $\sqrt{-1}$ 倍で置き換えてみると, $\mathbb{Z}\varpi + \mathbb{Z}\varpi\sqrt{-1}$ が $\sqrt{-1}$ 倍でかわらないことから,

$$\wp(\sqrt{-1}z) = -\wp(z)$$

が成り立つ. また, 微分方程式

$$\wp'(z)^2 = 4\wp(z)^3 - 4\wp(z)$$

さらにこれを微分した

$$\wp''(z) = 6\wp(z)^2 - 2$$

を満たす.

$\wp(z)$ の原点でのローラン展開を考える. 定義からそれが $\frac{1}{z^2}$ から始まることがわかるが, そのほかの係数を

$$\wp(z) = \frac{1}{z^2} + \sum_{n=2}^{\infty} \frac{2^n H_n}{n} \frac{z^{n-2}}{(n-2)!}$$

のように書く. すると上記 $\wp(-z) = \wp(z)$, $\wp(\sqrt{-1}z) = -\wp(z)$ より, H_n は n が 4 の倍数のとき以外は零であることがわかる. このときの H_n がフルヴィッツ数と呼ばれるものである. 微分方程式で$\wp''(z) = 6\wp(z)^2 - 2$ の展開係数を比較することにより,

$$H_4 = \frac{1}{10}$$

および, $n \geq 2$ に対し漸化式

$$(2n-3)(4n-1)(4n+1)H_{4n} = 3\sum_{i=1}^{n-1}(4i-1)(4n-4i-1)\binom{4n}{4i} H_{4i}H_{4(n-i)}$$

が得られる. これから, H_n は正の有理数であることがわかる. いくつかの値を表にする.

n	4	8	12	16	20	24	28	32
H_n	$\frac{1}{10}$	$\frac{3}{10}$	$\frac{567}{130}$	$\frac{43659}{170}$	$\frac{392931}{10}$	$\frac{1724574159}{130}$	$\frac{2498907956391}{290}$	$\frac{1671769422825579}{170}$

Hurwitz はこの H_n に対してクラウゼン・フォンシュタウトの定理の類似を証明した. それを述べるために次のフェルマー・オイラーの定理が必要になる.

定理 2.3 (フェルマー・オイラーの定理)**.**
4 で割って 1 余る素数は平方数の和として 1 通りに表される.

今 p を 4 で割ると 1 余る素数とし, その平方和への分解を

$$p = a^2 + b^2$$

とする. このとき a, b のいずれか一方のみが奇数でなくてはならないからそれを a とし, その符号を

$$a \equiv b + 1 \mod 4$$

で定める. このように定まる a を, a_p と書くことにする. このときフルヴィッツの定理は次のとおり.

定理 2.4 (フルヴィッツの定理)**.** 各 $n \geq 1$ に対してある整数 G_n があって,

$$H_{4n} = \frac{1}{2} + \sum_{\substack{p \equiv 1 \mod 4 \\ p-1 | 4n}} \frac{(2a_p)^{\frac{4n}{p-1}}}{p} + G_n.$$

p は 4 で割ると 1 余る素数で $p - 1$ が $4n$ を割るようなものを動く.
例として,

$$H_4 = \frac{1}{10} = \frac{1}{2} - \frac{2}{5},$$

$$H_8 = \frac{3}{10} = \frac{1}{2} + \frac{2^2}{5} - 1,$$

$$H_{12} = \frac{567}{130} = \frac{1}{2} - \frac{2^3}{5} + \frac{6}{13} + 5,$$

$$H_{16} = \frac{43659}{170} = \frac{1}{2} + \frac{2^4}{5} + \frac{2}{17} + 253.$$

この定理よりフルヴィッツ数 H_{4n} の分母は

$$2 \prod_{\substack{p \equiv 1 \mod 4 \\ p-1 | 4n}} p$$

とかけることがわかる.

また, ベルヌーイ数もフルヴィッツ数も関数の展開係数となる点や, ゼータ関数の特殊値になる点などにおいて類似性があり, 次の表のようになる.

	H_n	B_n
関数の展開係数	$\wp(z)=\dfrac{1}{z^2}+\displaystyle\sum_{n=2}^{\infty}\dfrac{2^nH_n}{n}\dfrac{z^{n-2}}{(n-2)!}$	$\dfrac{1}{\sin^2 x}=\dfrac{1}{x^2}+\displaystyle\sum_{n=2}^{\infty}\dfrac{(-1)^{\frac{n}{2}-1}2^nB_n}{n}\dfrac{x^{n-2}}{(n-2)!}$
ζ 関数の特殊値	$\displaystyle\sum_{\substack{\lambda\in\mathbb{Z}+\mathbb{Z}\sqrt{-1}\\ \lambda\neq 0}}\dfrac{1}{\lambda^n}=\dfrac{(2\varpi)^n}{n!}H_n$	$\displaystyle\sum_{r\in\mathbb{Z}-\{0\}}\dfrac{1}{r^n}=(-1)^{\frac{n}{2}-1}\dfrac{(2\pi)^nB_n}{n!}$
分母が求められる	$2\displaystyle\prod_{\substack{p\equiv 1 \mod 4\\ p-1\mid n}}p$	$\displaystyle\prod_{p-1\mid n}p$

3　ベルヌーイ数を計算するアルゴリズム

まずこのセクションでは, Fillebrown が [6] にて出した結果であるベルヌーイ数を計算するアルゴリズムについて述べる. 以降 $\overline{x}$ は x を丸め計算したものとする. また, $\langle x\rangle$ は x に最も近い整数とする. $G(n)=\lceil 2n\log_2 n\rceil$ とおくことにする.

アルゴリズム 1. もしすべての計算が $G(n)$ ビットで丸めが行われるならば以降のアルゴリズムで B_{2k} が計算される.

1. $2n+1$ 以下の素数 p を列挙する.
2. $\overline{r}_n=2(2n)!/(2\pi)^{2n}$ とする.
3. $\overline{t}_n=\prod_{p\leq\lceil 2n/\pi e\rceil}p^{2n}/(p^{2n}-1)$ とする.
4. $D_{2n}=\prod_{(p-1)\mid 2n}p$ とする.
5. $\overline{N}_{2n}=\overline{r}_n\overline{t}_nD_{2n}$ とする.
6. このとき $B_{2n}=(-1)^{n+1}\langle\overline{N}_{2n}\rangle/D_{2n}$ となる.

このアルゴリズムは B_{2n} を $O(n\log n)$ ビットの乗算 $O(n)$ 回で計算する.

このアルゴリズムを基にフルヴィッツ数を計算するアルゴリズムを次のようにして構成する.

4　フルヴィッツ数を計算するアルゴリズム

4.1　アルゴリズム

アルゴリズム 1 を基にして, 次のようなアルゴリズムによりフルヴィッツ数を計算することができる.

アルゴリズム 2. 以下のようにしてフルヴィッツ数 H_k を計算する. ($k\in 4\mathbb{N}$)

1. (準備)
   $k+1$ 以下の素数 p を列挙する.

$\alpha = \lceil (k+3)\log_2 k - 1.88k + 5.27 \rceil$
$M_k = \lceil 2\log((\frac{\pi}{\varpi})^k k^4 2^{\frac{k}{2}+5}\sqrt{(2k-2)!}) \rceil$
$C' = 2M_k + 2\log_2 k + 2k + 7$
$\beta = \lceil \alpha + 3 + \log_2 C' \rceil$
以降の計算は β ビットの精度で計算する.

2. (フルヴィッツ数を小数で計算する.)
$\overline{H(k)} = -\left(\frac{\pi}{\varpi}\right)^k \left(B_k - 2k\sum_{n=1}^{M_k}\frac{n^{k-1}}{e^{2\pi n}-1}\right)$
3. (フルヴィッツ数の分母を計算する.)
$D_k = 2\prod_{\substack{p\equiv 1 \mod 4\\ p-1|k}} p$
4. (フルヴィッツ数の分子を計算する.)
$\overline{N_k} = \overline{H(k)}D_k$
5. (有理数としてフルヴィッツ数を計算する.)
$H_k = \frac{\langle \overline{N_k}\rangle}{D_k}$

4.2 アルゴリズムの正しさの証明

まず, フルヴィッツ数を小数で表した $H(k)$ が step 2 のように求められることについて述べる. まず, 周期が 1 と i であるアイゼンシュタイン級数 $E_k(i)$ には次の等式が成り立つことが知られている [3, p.266].

$$\begin{aligned}E_k(i) &= \sum_{(m,n)\in\mathbb{Z}\times\mathbb{Z}-\{(0,0)\}} \frac{1}{(m+ni)^k}\\ &= (-1)^{\frac{k}{2}-1}\frac{(2\pi)^k}{k!}\left(B_k - 2k\sum_{n\geq 1}\frac{n^{k-1}}{e^{2\pi n}-1}\right).\end{aligned}$$

さらに $E_k(i)$ には次の等式が成り立つことが知られている [1, p.196].

$$E_k(i) = \frac{H_k}{k!}(2\varpi)^k.$$

これらの式をあわせると $k \in 4\mathbb{N}$ のとき,

$$H_k = -\left(\frac{\pi}{\varpi}\right)^k \left(B_k - 2k\sum_{n\geq 1}\frac{n^{k-1}}{e^{2\pi n}-1}\right).$$

次にフルヴィッツ数の分母 D_k だがこれは第 2 節で述べたとおりフルヴィッツの定理を用いると,

$$D_k = 2\prod_{\substack{p\equiv 1 \mod 4\\ p-1|k}} p$$

と表される．また，$\displaystyle\prod_{p\leq x} p \leq 4^x$ という事実 [9, p.183] を用いると，

$$
\begin{aligned}
D_k &\leq (k+1)(\frac{k}{2}+1)(\frac{k}{3}+1)\prod_{p\leq \frac{k}{4}+1} p \\
&\leq (k+1)(\frac{k}{2}+1)(\frac{k}{3}+1)4^{\frac{k}{4}+1} \qquad (1)
\end{aligned}
$$

となる．さらに，

$$
\begin{aligned}
\left|\sum_{(m,n)\in\mathbb{Z}\times\mathbb{Z}-\{(0,0)\}} \frac{1}{(m+ni)^k}\right| &\leq 8\zeta(3) \\
&= 9.61645\cdots
\end{aligned}
$$

という事実 [5, p.112] を用いると，フルヴィッツ数の分子 N_k は

$$
\begin{aligned}
N_k &= D_k H_k \\
&= \frac{k!}{(2\varpi)^k} D_k \sum_{(m,n)\in\mathbb{Z}\times\mathbb{Z}-\{(0,0)\}} \frac{1}{(m+ni)^k} \\
&\leq (\frac{k}{5.24})^k (k+1)(\frac{k}{2}+1)(\frac{k}{3}+1)(\prod_{p\leq \frac{k}{4}+1} p) 8\zeta(3) \\
&\leq (\frac{k}{5.24})^k (9.62) k^3 4^{\frac{k}{4}+1}
\end{aligned}
$$

と書けるので，

$$
\begin{aligned}
\log_2 N_k &\leq k(\log_2 k - \log_2 5.24) + \log_2 9.62 + 3\log_2 k + \frac{k}{2} + 2 \\
&\leq (k+3)\log_2 k - 1.88k + 5.27
\end{aligned}
$$

が成り立つ．よって

$$
\alpha = \lceil (k+3)\log_2 k - 1.88k + 5.27 \rceil
$$

とすると

$$
N_k \leq 2^\alpha
$$

となる．このようにして分子のサイズを決定することが可能である．次に，

$$
\tilde{N_k} = -\left(\frac{\pi}{\varpi}\right)^k D_k \left(B_k - 2k\sum_{n=1}^{M_k} \frac{n^{k-1}}{e^{2\pi n}-1}\right)
$$

とおくことにする．このアルゴリズムにおいて $\tilde{N_k}$ を β ビットの精度で計算したものを $\overline{N_k}$ とする．ここで，$N_k = \langle \overline{N_k} \rangle$ であることを示す．そのために $0 < N_k - \tilde{N_k} < \frac{1}{4}$ と $|\tilde{N_k} - \overline{N_k}| < \frac{1}{4}$ を示す．この二つが成立するならば，$|N_k - \overline{N_k}| < \frac{1}{2}$ となり，$N_k = \langle \overline{N_k} \rangle$ であることが示される．

定理 4.1. 上記のようにして $\tilde{N}_k$ を定めると, $0 < N_k - \tilde{N}_k < \frac{1}{4}$ となる.
上記の定理の証明のために次の補題を示す.

補題 4.2.

$$\sum_{n=M_k+1}^{\infty} \frac{n^{k-1}}{e^{2\pi n}-1} < ke^{-\frac{M_k}{2}}\sqrt{(2k-2)!}.$$

(証明) まず,

$$\left(\frac{x^{k-1}}{e^x}\right)' = \frac{x^{k-2}(k-1-x)}{e^x}$$

これの正負を考える. $M_k+1 \leq x$ となるように x を取ると, $k-1 \leq M_k$ だから

$$\left(\frac{x^{k-1}}{e^x}\right)' < 0$$

そのため, $\dfrac{x^{k-1}}{e^x}$ は単調減少となる.

$$\begin{aligned}\sum_{n=M_k+1}^{\infty} \frac{n^{k-1}}{e^{2\pi n}-1} &< \sum_{n=M_k+1}^{\infty} \frac{n^{k-1}}{e^n} \\ &< \int_{M_k}^{\infty} \frac{x^{k-1}}{e^x}dx \\ &= e^{-M_k}(M_k^{k-1} + (k-1)M_k^{k-2} + \cdots\cdots + (k-1)!M_k + (k-1)!) \\ &< e^{-M_k}kM_k^{k-1}\end{aligned}$$

となる. さらに,

$$\begin{aligned}\sqrt{e^{M_k}} &= \sqrt{\sum_{l=0}^{\infty} \frac{M_k^l}{l!}} \\ &> \sqrt{\frac{M_k^{2(k-1)}}{(2k-2)!}} \\ &= \frac{M_k^{k-1}}{\sqrt{(2k-2)!}}\end{aligned}$$

であるため,

$$\begin{aligned}\sum_{n=M_k+1}^{\infty} \frac{n^{k-1}}{e^{2\pi n}-1} &< e^{-M_k}kM_k^{k-1} \\ &< ke^{-\frac{M_k}{2}}\sqrt{(2k-2)!}\end{aligned}$$

となり, 補題の証明が完了する. (証明終)

続いて, 定理の証明を行う.

(証明)　(1) および, $(k+1)(\frac{k}{2}+1)(\frac{k}{3}+1) \leq k^3$ より

$$
\begin{aligned}
N_k - \tilde{N_k} &= \left(\frac{\pi}{\varpi}\right)^k D_k \left(2k \sum_{M_k+1}^{\infty} \frac{n^{k-1}}{e^{2\pi n}-1}\right) \\
&< \left(\frac{\pi}{\varpi}\right)^k (2ke^{-\frac{M_k}{2}}\sqrt{(2k-2)!})k^3 4^{\frac{k}{4}+1}
\end{aligned}
$$

である. また M_k の取り方から,

$$
2\log\left(\left(\frac{\pi}{\varpi}\right)^k k^4 2^{\frac{k}{2}+5}\sqrt{(2k-2)!}\right) < M_k
$$

$$
\left(\frac{\pi}{\varpi}\right)^k k^4 2^{\frac{k}{2}+5}\sqrt{(2k-2)!} < e^{\frac{M_k}{2}}
$$

よって, $0 < N_k - \tilde{N_k} < \frac{1}{4}$ となり証明が完了する. (証明終)

続いて次の補題が成り立つことが [7] より分かる.

補題 4.3. [7, Lemma 3.1] $i=1$ から n に対して$|\delta_i| \leq u,\ \rho_i = \pm 1,\ nu < 1$ とすると,

$$
\prod_{i=1}^{n}(1+\delta_i)^{\rho_i} = 1+\theta_n,\ (|\theta_n| \leq \frac{nu}{1-nu} = \gamma_n)
$$

を満たす θ_n が存在する.

これを用いると次の定理が示せる.

定理 4.4. 上記のようにして $\bar{N_k}$ を定めると, $|\overline{N_k} - \tilde{N_k}| < \frac{1}{4}$ となる.

(証明) $\overline{x}$ は実数 x を β ビットで丸めたものとする. [7] によると $\bar{x} = x(1+\delta)$, $(|\delta| \leq u = 2^{-\beta})$ とかける. この性質を用いて $\overline{N_k}$ を計算する. 以降 λ, ϵ, μ に対しても δ と同様の仮定をする. まず,

$$
\overline{e^{2\pi}} = e^{2\pi}(1+\delta), |\delta| \leq u
$$

とする. また, e のべき乗は

$$
\overline{e^{2\pi n}} = \overline{\overline{e^{2\pi}} \cdot \overline{e^{2\pi(n-1)}}}
$$

で計算される. そのため帰納法を用いると次のことが分かる.

$$
\begin{aligned}
\overline{e^{2\pi n}} &= e^{2\pi n}(1+\delta)^n \prod_{i=1}^{n-1}(1+\delta_i) \\
&= e^{2\pi n}(1+\theta_{2n-1}).
\end{aligned}
$$

このようにして θ_{2n-1} を定めると, $|\theta_{2n-1}| \leq \gamma_{2n-1}$ となる.

$$
\overline{e^{2\pi n}} - 1 = (e^{2\pi n}-1)(1+\theta_{2n-1})(1+\delta_{2n})
$$

で δ_{2n} を定める．このとき

$$\begin{aligned}|\delta_{2n}| &= \frac{|\theta_{2n-1}|}{(e^{2\pi n}-1)(1+\theta_{2n-1})} \\ &\le \frac{(2n-1)u}{(e^{2\pi n}-1)(1-(4n-2)u)}.\end{aligned}$$

このとき M_k, α の定め方より

$$\begin{aligned}4M_k u &\le 2^{-\alpha-2} \\ &\le \frac{1}{4}\end{aligned}$$

であるので

$$1-(4n-2)u \ge \frac{3}{4}$$

より

$$\frac{(2n-1)u}{(e^{2\pi n}-1)(1-(4n-2)u)} \le \frac{4}{3}\frac{2n-1}{e^{2\pi n}-1}u$$

が成り立つ．また $n \ge 1$ のとき

$$\frac{2n-1}{e^{2\pi n}-1} \le \frac{3}{4}$$

であるので, $|\delta_{2n}| \le u$ となる．以上のことより $\overline{\overline{e^{2\pi n}}-1} = (\overline{e^{2\pi n}}-1)(1+\delta_{2n+1})$ のように δ_{2n+1} を定めると, 次の式が成り立つ．

$$\begin{aligned}\overline{\left(\frac{n^{k-1}}{e^{2\pi n}-1}\right)} &= \frac{n^{k-1}}{e^{2\pi n}-1}\cdot\frac{1+\delta_{2n+2}}{(1+\theta_{2n-1})(1+\delta_{2n})(1+\delta_{2n+1})} \\ &= \frac{n^{k-1}}{e^{2\pi n}-1}(1+\theta_{2n+2}).\end{aligned}$$

このような $\theta_{2n+2}(|\theta_{2n+2}| \le \gamma_{2n+2})$ が存在することが補題 4.3 より分かる．

そのため

$$a_n = \frac{n^{k-1}}{e^{2\pi n}-1}$$

とおくと

$$\overline{a_n} = a_n(1+\theta_{2n+2}^{(n)})$$

がわかった $(|\theta_{2n+2}^{(n)}| \le \gamma_{2n+2})$.

$$S_m = \sum_{n=1}^{m} a_n$$

とすると

$$\overline{S_m} = S_m(1+\eta_{2m+3}), \quad |\eta_{2m+3}| \le \gamma_{2m+3}$$

となる η_{2m+3} が存在する．これは帰納法により次のように示される．$m=1$ のときは明らかである．一般の m に対して $m-1$ までは成り立っているとすると

$$
\begin{aligned}
\overline{S_m} &= \overline{\overline{S_{m-1}}+\overline{a_m}} \\
&= \overline{S_{m-1}(1+\eta_{2m+1})+a_m(1+\theta^{(m)}_{2m+2})} \\
&= (S_{m-1}(1+\eta_{2m+1})+a_m(1+\theta^{(m)}_{2m+2}))(1+\delta).
\end{aligned}
$$

ここで

$$
\begin{aligned}
|S_{m-1}(1+\eta_{2m+1})+a_m(1+\theta^{(m)}_{2m+2})-(S_{m-1}+a_m)| &= |S_{m-1}\eta_{2m+1}+a_m\theta^{(m)}_{2m+2}| \\
&= S_{m-1}|\eta_{2m+1}|+a_m|\theta^{(m)}_{2m+2}| \\
&\le S_{m-1}\gamma_{2m+1}+a_m\gamma_{2m+2} \\
&\le (S_{m-1}+a_m)\gamma_{2m+2}
\end{aligned}
$$

であるため，$S_{m-1}(1+\eta_{2m+1})+a_m(1+\theta^{(m)}_{2m+2})=(S_{m-1}+a_m)(1+\eta_{2m+2})$ で η_{2m+2} を定めると $|\eta_{2m+2}|\le\gamma_{2m+2}$ が成り立つ．よって，

$$
\begin{aligned}
\overline{S_m} &= S_m(1+\eta_{2m+2})(1+\delta) \\
&= S_m(1+\eta_{2m+3})
\end{aligned}
$$

で η_{2m+3} を定めると，

$$
\begin{aligned}
|\eta_{2m+3}| &= |(1+\eta_{2m+2})(1+\delta)-1| \\
&= |\eta_{2m+2}+\delta+\eta_{2m+2}\delta| \\
&\le |\eta_{2m+2}|+|\delta|+|\eta_{2m+2}||\delta| \\
&\le \gamma_{2m+2}+u+\gamma_{2m+2}u \\
&= \frac{(2m+2)u}{1-(2m+2)u}+u+\frac{(2m+2)u^2}{1-(2m+2)u} \\
&= \frac{(2m+3)u}{1-(2m+2)u} \\
&\le \frac{(2m+3)u}{1-(2m+3)u} \\
&= \gamma_{2m+3}.
\end{aligned}
$$

よって全ての自然数 m に対して成り立つことが分かる．続いて

$$\overline{B_k}=B_k(1+\lambda_k)$$

とする．さらに，

$$\overline{\pi}=\pi(1+\mu'),$$

$$\overline{\varpi}=\varpi(1+\mu),$$

$$\overline{\left(\frac{\overline{\pi}}{\overline{\varpi}}\right)^k}=\left(\frac{\pi}{\varpi}\right)^k\frac{(1+\mu')^k}{(1+\mu)^k}\prod_{i=1}^{r_k}(1+\epsilon_i)$$

とかける. r_k は k 乗を計算するのに必要な乗算の回数であり, $r_k \le 2\log_2 k$ である. 以上より,

$$
\begin{aligned}
\overline{H(k)} &= -\overline{\left(\frac{\overline{\pi}}{\overline{\varpi}}\right)^k \overline{(\overline{B_k} - \overline{2k\overline{S_{M_k}}})}} \\
&= -\left(\frac{\pi}{\varpi}\right)^k (B_k - 2kS_{M_k})\frac{(1+\mu')^k}{(1+\mu)^k}\prod_{i=1}^{r_k}(1+\epsilon_i) \\
&\quad \cdot\ (1+\eta_{2M_k+3})(1+\delta_{2M_k+4})(1+\delta_{2M_k+5})(1+\delta_{2M_k+6}).
\end{aligned}
$$

これより,

$$
\begin{aligned}
C &= k + k + r_k + 2M_k + 6 \\
&= 2M_k + r_k + 2k + 6
\end{aligned}
$$

とする. これは, $Cu < 1$ を満たす.

$$
\overline{H(k)} = H(k)(1+\theta_C)
$$

で θ_C を定めると

$$
|\theta_C| \le \gamma_C = \frac{Cu}{1-Cu}
$$

を満たす. このことから,

$$
\begin{aligned}
\overline{N_k} &= \overline{\overline{H(k)}D_k} \\
&= H(k)D_k(1+\delta)(1+\theta_C) \\
&= H(k)D_k(1+\theta_{C+1}) \\
&= \tilde{N}_k(1+\theta_{C+1})
\end{aligned}
$$

が成り立つような θ_{C+1} をとると $|\theta_{C+1}| \le \gamma_{C+1}$ を満たす. また, $C' = 2M_k + 2\log_2 k + 2k + 7$ とおくと

$$
\begin{aligned}
|\theta_{C+1}| &\le \gamma_{C+1} \\
&\le \gamma_{C'} \\
&= \frac{C'u}{1-C'u}\ (\text{また, } u = 2^{-\beta}).
\end{aligned}
$$

以上より β の取り方より, $C'u < 1/2$ となり次が成り立つ.

$$
\begin{aligned}
|\overline{N_k} - \tilde{N}_k| &= \tilde{N}_k|\theta_{C+1}| \\
&\le \tilde{N}_k\gamma_{C'} \\
&= \tilde{N}_k\frac{C'u}{1-C'u} \\
&< 2\tilde{N}_kC'u \\
&< 2(2^\alpha)C'2^{-\beta}.
\end{aligned}
$$

また, $\beta = \lceil \alpha + 3 + \log_2 C' \rceil$ であるため,

$$2^{-\beta} \leq 2^{-\alpha-3} C'^{-1}$$

を満たすので,

$$\begin{aligned} |\overline{N_k} - \tilde{N_k}| &< 2(2^{\alpha}) C' 2^{-\beta} \\ &\leq \frac{1}{4} \end{aligned}$$

となるので証明が完了する. (証明終)

5 アルゴリズムの計算量

このセクションで前節で与えられたアルゴリズムがどれほどの計算量で H_n を計算するかについて述べる. また, このセクションでは n ビットの 2 つの数の乗算の計算量 $M(n)$ は, $M(n) = O(n \log n \log\log n)$ で与えられ, さらに n ビットの数と m ビットの数 $(n > m)$ の乗算の計算量 $M(n, m)$ は, $M(n, m) = O(n \log m \log\log m)$ と書くことができる. これは Schönhage と Strassen のアルゴリズムを使うことで分かる [8]. 次に下記の定理を示す.

定理 5.1. アルゴリズム 2 は H_k をビット演算量 $O(M(k \log k) k (\log k)^2)$ で計算する.

(証明) まず, $\beta = O(k \log k), M_k = O(k \log k)$ であることが β, M_k のとり方から分かる. step 2 以降は β ビットの精度で計算するときの計算量について考える. また, [2, Chapter 7] を参照すると, π は ビット演算量 $O(M(k \log k) \log k)$ で計算され, $e^{2\pi}$ はビット演算量 $O(M(k \log k) \log k)$ で計算され, ϖ はビット演算量 $O(M(k \log k) \log k)$ で計算されることが分かる. 続いてアルゴリズムの各ステップにおける計算量を見ていく.

step 1(準備)
$k+1$ 以下の素数の列挙のビット演算量は [4, Algorithm 3.2.2] によると $O(k \log k / \log\log k)$ である. α, β, M_k, C' の計算量は小さいので無視する.

step 2(フルヴィッツ数を小数で計算する.)
$\left(\dfrac{\pi}{\varpi}\right)^k$ についてみていく. これは β ビットの数の乗算を高々 $2 \log_2 k$ 回, 除算を 1 回行う. そのため, ビット演算量は $O(\log k \cdot M(k \log k))$ となる.
次に, B_k の計算量だが [6] によると, $O(k \log k)$ ビット数の乗算を $O(k)$ 回繰り返す. そのため, ビット演算量は $O(M(k \log k) k)$ である.
n^{k-1} の計算は一回で $O(M(k \log k) \log k)$ で計算でき, それを $M_k - 1$ 回繰り返すので, ビット演算量は $O(M_k M(k \log k) \log k)$ となる. また, $e^{2\pi n} = e^{2\pi} \cdot e^{2\pi(n-1)}$ と計算される. そのため, $e^{2\pi n}$ は一回で $O(M(k \log k))$ で計算でき, それを $M_k - 1$ 回繰り返すので, ビット演算量は $O(M_k M(k \log k))$ となる. $\frac{n^{k-1}}{(e^{2\pi n} - 1)}$ における除算 M_k 回分の計算量は $O(M_k M(k \log k))$ になる. それ以外の計算量は無視できる. よって step 2 全体のビット演算量は, $O(M_k M(k \log k) \log k)$ である.

step 3(分母の計算)

$D_k = 2 \prod_{\substack{p \equiv 1 \mod 4 \\ p-1|k}} p$ の計算量を考える．素数定理より k 以下の素数の個数は $O(\frac{k}{\log k})$ である．乗算一回につきビット演算量は $O(M(k, \log k))$ であり，これを $\frac{k}{\log k}$ 回繰り返すので求めるビット演算量は $O(\frac{k}{\log k} M(k, \log k))$ となる．

step 4

$N_k = H(k)D(k)$ の計算量を考える．$H(k)$ は $O(k \log k)$ ビットで，$D(k)$ は $O(k)$ ビットであるので，ビット演算量 $O(M(k \log k, k))$ で計算する．

step 5

最も近い整数を計算するビット演算量は $O(k \log k)$ である．

以上より最も計算量が多いのは step 2 であることが分かる．よってアルゴリズム 2 は H_k をビット演算量 $O(M(k \log k) M_k \log k) = O(M(k \log k) k (\log k)^2)$ で計算する．(証明終)

続いてフルヴィッツ数を漸化式で計算するときの計算量について述べる．

定理 5.2. フルヴィッツ数を漸化式

$$(2k-3)(4k-1)(4k+1)H_{4k} = 3 \sum_{i=1}^{k-1} (4i-1)(4k-4i-1) \binom{4k}{4i} H_{4i} H_{4(k-i)}$$

を用いて計算するときビット演算量は $O(M(k \log k) k^2 \log k)$ かかる．

(証明) 漸化式を変形すると次のように書ける．

$$H_{4k} = \frac{3}{(2k-3)(4k-1)(4k+1)} \sum_{i=1}^{k-1} (4i-1)(4k-4i-1) \binom{4k}{4i} H_{4i} H_{4(k-i)}$$

この計算量を考えるために次の補題を用いる．

補題 5.3. 分子，分母が高々 k ビットの有理数の四則演算のビット演算量は $O(M(k) \log k)$ である．
(証明) まず，有理数の加算にかかるビット演算量を考える．a, b, c, d を高々 k ビット とする．

$$\frac{b}{a} + \frac{d}{c} = \frac{bc + ad}{ac}$$

これは乗算 3 回，加算 1 回，最大公約数の計算が 1 回，除算 2 回行う．そのなかでは最大公約数の計算がもっとも計算量が多く，ビット演算量 $O(M(k) \log k)$ で有理数の加算を一度行える．続いて有理数の乗算について考えると，乗算 2 回，最大公約数の計算を 1 回，除算 2 回行う．よって乗算にかかるビット演算量は $O(M(k) \log k)$ である．(証明終)

定理の証明に戻る．2 項係数 $\binom{k}{l}$ の計算量について見る．$\binom{k}{l} \leq 2^k$ より $\binom{k}{l}$ のサイズは $O(k)$ ビットである．パスカルの三角形を用い，2 項係数 $\binom{a}{b} (0 \leq b \leq a \leq 4k)$ を全て計算する．よって漸化式 $O(k)$ 回全部の分のビット演算量は $O(k^3)$ である．

$H_{4i} H_{4(k-i)}$ はサイズが $O(k \log k)$ ビットの有理数の乗算であるためビット演算量は

$O(M(k\log k)\log k)$ である．これを $O(k)$ 回繰り返すのでビット演算量は $O(M(k\log k)k\log k)$ となる．

漸化式の和の部分は分子分母のサイズが $O(k\log k)$ ビット となっている有理数の和である．加算 1 回につきビット演算量は $O(M(k\log k)\log k)$ であり加算を $O(k)$ 回繰り返すので漸化式 1 回のビット演算量は, $O(M(k\log k)k\log k)$ となる．よって全体では漸化式を $O(k)$ 回繰り返すので全体のビット演算量は $O(M(k\log k)k^2\log k)$ となる．これは $O(k^3)$ よりも大きい．

以上よりフルヴィッツ数を漸化式

$$(2k-3)(4k-1)(4k+1)H_{4k} = 3\sum_{i=1}^{k-1}(4i-1)(4k-4i-1)\binom{4k}{4i}H_{4i}H_{4(k-i)}$$

を用いて計算するときビット演算量は $O(M(k\log k)k^2\log k)$ かかる．(証明終)

6 数値実験

アルゴリズム 2 と定理 5.2 の漸化式それぞれを用いて計算速度の比較を行った．利用した計算機は Intel Core i5-4210M 2.60GHz のプロセッサ, メモリ 8.00GB を実装した Windows7, 64bit 版のシステム, ソフトウェアは PARI/GP 2.7.3 [10] である．

6.1 数値実験結果

数値実験の結果は次のようになった．(単位:msec)

	H_{80}	H_{400}	H_{1000}	H_{2000}	H_{4000}	H_{6000}	H_{8000}
アルゴリズム 2	31	31	438	2714	16349	48001	1min43418
漸化式	16	93	1467	6505	1min23960	6min30828	18min21502

7 まとめ

本論文ではフルヴィッツの定理などを用いてフルヴィッツ数を計算するアルゴリズムを構成した．実験結果から十分大きい自然数 k に対して H_k を計算するスピードはアルゴリズム 2 の方が速いことが分かった．より計算量の少ないアルゴリズムを構成すること, また $\mathbb{Z}+\mathbb{Z}\sqrt{-1}$ 以外の格子に対応するフルヴィッツ数への拡張などが今後の課題である．

8 謝辞

本研究は, 著者が首都大学東京大学院理工学研究科数理情報科学専攻博士前期課程在学中に, 同大学院理工学研究科数理情報科学専攻の内田幸寛准教授の指導のもとに行ったものである．適切な助言を賜り, 熱心に指導して下さった内田幸寛准教授に深く感謝致します．またご多忙の中, 本論文の副査を快諾していただきました内山成憲教授と津村博文教授に感謝致します．

参考文献

[1] 荒川恒男・伊吹山知義・金子昌信 『ベルヌーイ数とゼータ関数』(牧野書店, 2001)

[2] J. M. Borwein, P. B. Borwein. *Pi and the AGM: A Study in Analytic Number Theory and Computational Complexity* (John Wiley, New York, 1987)

[3] H. Cohen. *Number Theory Volume II: Analytic and Modern Tools* (Springer, New York, 2007)

[4] R. Crandall, C. Pomerance. *Prime Numbers: A Computional Perspective* (2nd ed., Springer, New York, 2005)

[5] 土井公二・三宅敏恒 『保型形式と整数論』(紀伊国屋書店, 1976)

[6] S. Fillebrown. Faster Computation of Bernoulli Numbers. J. Algorithms **13**, (1992), 431–445.

[7] N. J. Higham. *Accuracy and Stability of Numerical Algorithms* （John Wiley, New York, 1987）

[8] D. E. Knuth. *The Art of Computer Programming. Vol.2: Seminumerical Algorithms* (3rd ed., Reading, MA, Mass Addison-Wesley, 1997)

[9] L. Niven, H. S. Zuckerman. *An Introduction to the Theory of Numbers* (John Wiley, New York, 1960)

[10] The PARI Group, PARI/GP version 2.7.3, Bordeaux, 2015, http://pari.math.u-bordeaux.fr/.

[11] W. Stein. *Modular Forms, a Computational Approach* (American Mathematical Society, Providence, RI, 2007)

付録 A　PARI/GP のプログラム

以下にフルヴィッツ数を計算する PARI/GP [10] のプログラムを載せる.

```
el(k)=
{my(w);
w = ellperiods([1,I]);
 elleisnum(w, k);
}


h(k)=
{my(b,e,l,E);
b=bernreal(k);
```

```
e=el(k);
E=ellinit([-1,0]);
l=E.omega[1];
(-e)*b/((2*l)^k);
};

d(k)=
{my(r,p);
r=1;forprime(p=2,k+1,if((p-1)%4==0,if(k%(p-1)==0,r*=p)));
2*r;
};
f(k)=
{my(oldprec,r,s,t,a,m,c,b);
s=d(k);
oldprec=default(realprecision);
a=ceil((k+3)*log(k)/log(2)-(1.88)*k+5.27);
m=ceil(2*(log(((Pi)/(2.62))^k*k^4*sqrt(2^(k+10))*sqrt((2*k-2)!))));
c=2*m+2*log(k)/log(2)+2*k+7;
b=ceil(a+3+log(c)/log(2));
default(realprecision,max(28,
ceil(
b*log(2)/log(10))
));
t=round(s*h(k));
r=t/s;
default(realprecision,oldprec);
r;
};
```

続いて漸化式を用いてフルヴィッツ数を計算するプログラムのソースを載せる.

```
hur(m)=
{my(h,n,i);h=vector(m);
h[1]=1/10;
for(n=2,m,h[n]=(3/((2*n-3)*(4*n-1)*(4*n+1)))*
(sum(i=1,n-1,(4*i-1)*(4*(n-i)-1)*binomial(4*n,4*i)*h[i]*h[n-i])));
h[m];}
```